VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ROZŠÍŘENÁ REALITA PRO DESKOVÉ HRY A VZDĚLÁVACÍ APLIKACE

AUGMENTED REALITY FOR DESK GAMES AND EDUCATIONAL APPLICATIONS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **JIŘÍ ZÁLESKÝ**
AUTHOR

VEDOUCÍ PRÁCE **Ing. VÍTĚSLAV BERAN, Ph.D.**
SUPERVISOR

BRNO 2018

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2017/2018

Zadání bakalářské práce

Řešitel: **Záleský Jiří**
Obor: Informační technologie
Téma: **Rozšířená realita pro deskové hry a vzdělávací aplikace**
Augmented Reality for Desk Games and Educational Applications
Kategorie: Uživatelská rozhraní

Pokyny:

1. Prostudujte problematiku tvorby grafických uživatelských rozhraní a testování uživatelské zkušenosti. Seznamte se s problematikou rozšířené reality, tangible interfaces, existujícími herními enginy a vizualizačními nástroji.
2. Navrhněte systém pro tvorbu her pro interaktivní stůl (rozšířená realita promítaná na stůl). Vstupem systému jsou informace o situaci na stole (pozice a stav objektů, pozice a stav ruky uživatele). Systém musí umožňovat realizovat GUI hry a podporovat herní logiku.
3. Vyberte vhodné nástroje a navržený systém implementujte. Vstupy systému simulujte nebo využijte externí řešení.
4. Řešení demonstrujte na vybrané stolní zábavné nebo vzdělávací hře.
5. Proveďte testování na uživatelích a vyhodnoťte uživatelskou spokojenost.
6. Vytvořte plakát a krátké video prezentující klíčové výsledky vašeho řešení.

Literatura:

- B. Shneiderman, C. Plaisant. *Designing the user interface: strategies for effective human-computer interaction*, Addison-Wesley Computing, ISBN-10: 0-321-53735-1, 2009
- L. Mathis. *Designed for Use: Create Usable Interfaces for Applications and the Web*, Pragmatic Bookshelf, ISBN-10: 1934356751, 2011
- L. Shklar, R. Rosen. *Web Application Architecture: Principles, Protocols and Practices*, Wiley, ISBN-10: 047051860X, 2009

Pro udělení zápočtu za první semestr je požadováno:

- Body 1, 2, 3 a částečně bod 4.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Beran Vítězslav, Ing., Ph.D.**, UPGM FIT VUT
Datum zadání: 1. listopadu 2017
Datum odevzdání: 16. května 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Jan Černocký
vedoucí ústavu

Abstrakt

Tato bakalářská práce se věnuje tématu rozšířené reality a jejího užití pro stolní hry a výukové aplikace na interaktivním stole. Cílem bylo vytvoření samostatného systému umožňujícího snadné vytváření her pro interaktivní stůl, a to formou návrhu systému umožňujícího komunikaci s interaktivním stolem a vývojem demoaplikace, která prezentuje funkcionalitu systému. V části týkající se návrhu systému byly představeny v práci použité technologie a nástroje, např. C++, Qt nebo Catkin, a vytvořen první návrh hry - demoaplikace. V implementační části je vysvětlen způsob převodu návrhu na fungující aplikaci, včetně popisu vnitřní struktury aplikace. Poslední část práce, věnovaná testování, přináší kromě odhalení chyb aplikace i další návrhy na její rozšíření v budoucnu. Výsledkem práce je fungující systém pro rozšířenou realitu a demoaplikace na něho navazující, což v budoucnu poslouží k vytváření dalších programů využívajících interaktivní stůl. Systém je určený pro použití na interaktivním stole ARTable3.

Abstract

This bachelor thesis deals with the topic of augmented reality and its use for board games and educational applications on the interactive table. The goal was to create a stand-alone system for an easy interactive game table creation, by designing a system that allows communication with the interactive table and the creation of a demo-application that presents the functionality of the system. In the system design section, the technologies and tools used, such as C ++, Qt, or Catkin, were introduced and the first design of the game - demo-application was created. The implementation section explains how to transform the design into a functioning application, and includes a description of the internal structure of the application. The last part of the work focused on testing brings, in addition to revealing the application errors, other suggestions for its expansion in the future. The result of the work is a functioning system for augmented reality and demo-application for it, which in the future will help to create other programs for an interactive table. The system is designed for use on the interactive ARTable3 table.

Klíčová slova

Rozšířená realita, interaktivní stůl, ARTable3, OpenCV, Homografie, ROS, uživatelská rozhraní, herní enginy

Keywords

Augmented Reality, interactive table, ARTable3, OpenCV, Homography, ROS, user interfaces, game engines

Citace

ZÁLESKÝ, Jiří. *Rozšířená realita pro deskové hry a vzdělávací aplikace.* Brno, 2018. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Vítěslav Beran, Ph.D.

Rozšířená realita pro deskové hry a vzdělávací aplikace

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Vítězslava Berana Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Jiří Záleský
14. května 2018

Poděkování

Tímto bych chtěl poděkovat panu Ing. Vítězslavu Beranovi Ph.D. za odborné konzultace. Dále pak členům výzkumné skupiny Robo@FIT za odborné informace a konzultace ohledně interaktivního stolu a jeho náležitostí.

Obsah

Kapitola 1

Úvod

V dnešní době je možné často narazit na téma rozšířené reality. Velkým lákadlem je hlavně pro mladé lidi, kteří jsou každodenně v kontaktu s elektronickými zařízeními. Toho lze využít nejen pro účely zábavy, ale také pro účely interaktivního vzdělávání.

Tématem projektu je systém pro rozšířenou realitu, který by umožnil tvorbu deskových her a výukových aplikací pro interaktivní stůl. Součástí je také vytvoření demonstrační aplikace, která by systém využívala. V textu lze nalézt návrh řešení projektu a jednotlivé postupy pro implementaci.

Prvním krokem pro dosažení cíle je seznámení se s tématy grafických uživatelských rozhraní, rozšířené reality a hmotných uživatelských rozhraní. K tomu patří nastudování a určení vizualizačních nástrojů vhodných pro řešení projektu. Informace z výzkumu těchto témat posloužily k vytvoření vhodného uživatelského prostředí. Dále bylo třeba se zaměřit na existující herní enginy, porovnat jejich vlastnosti a v případě potřeby některý z nich využít při tvorbě aplikace. Poslední z teoretických znalostí nutných pro vytvoření a ladění vytvořeného systému a aplikace bylo téma testování uživatelských zkušeností. Tyto nabyté znalosti jsou shrnuty v kapitole 2. Součástí této kapitoly jsou také nástroje vybrané pro implementaci.

Následujícím krokem je vytvoření návrhu systému a uživatelského rozhraní cílového programu na základě získaných znalostí. Nezbytnou součástí práce je nalezení či vymyšlení vhodné stolní hry nebo vzdělávací aplikace pro vytvoření demonstrační aplikace. Pro detailní vyvětlení hry je v práci napsán příkladový scénář toho, jak by mohla hra probíhat. Značná část návrhu se věnuje komponentám navržených pro hru. Součástí návrhu je i způsob interakce uživatele a systému skrze rozhraní promítané na stůl a snímání obrazu, grafické uživatelské rozhraní a herní logika. Návrh systému a aplikace je popsán v kapitole 3.

Kapitola 4 popisuje způsob, jakým byl návrh převeden do implementační roviny. Lze zde nalézt podrobné informace o implementaci grafického rozhraní a herní logiky hry. Další část se zabývá způsobem, který je použit pro komunikaci systému s interaktivním stolem a uživatelem. V závěru kapitoly jsou řešeny kritické body implementace.

Následující kapitola, tedy kapitola 5, se zabývá tématem testování. Nejprve jsou vysvětleny obecné principy testování, následované rozborem prováděných testování. V rozborech jsou uvedena data o testujících skupinách, použitých metodách i průbězích testování. V závěru každého testování je vyhodnocení výsledků zpětné vazby, údaje o nalezených nedostatcích a návrhy na jejich opravu a vylepšení aplikace.

Součástí práce jsou prvky pro prezentaci obsahu práce - plakát a demonstrační video, které jsou umístěny na přiloženém kompaktním disku.

Kapitola 2

Klíčové znalosti a technologie

Tato kapitola se zaobírá teoretickými znalostmi, které byly nutné pro vytvoření návrhu a realizaci implementace. Jsou zde uvedeny znalosti z uživatelských rozhraní, rozšířené reality, ale také počítačového vidění.

2.1 Uživatelská rozhraní

Uživatelské rozhraní, známé také pod názvem User Interface (UI) je prostor nebo způsob používaný pro interakci mezi určitým zařízením a člověkem. Hlavním cílem uživatelského rozhraní je komunikace se zařízením umožňujícím jednoduché zadávání vstupních informací a přehlednou interpretaci informací výstupních.

Uživatelská rozhraní se na základě reprezentace vstupních a výstupních údajů dělí do skupin. Tato práce je zaměřena hlavně na grafická a hmotná uživatelská rozhraní, přesto je vhodné uvést i další příklady.

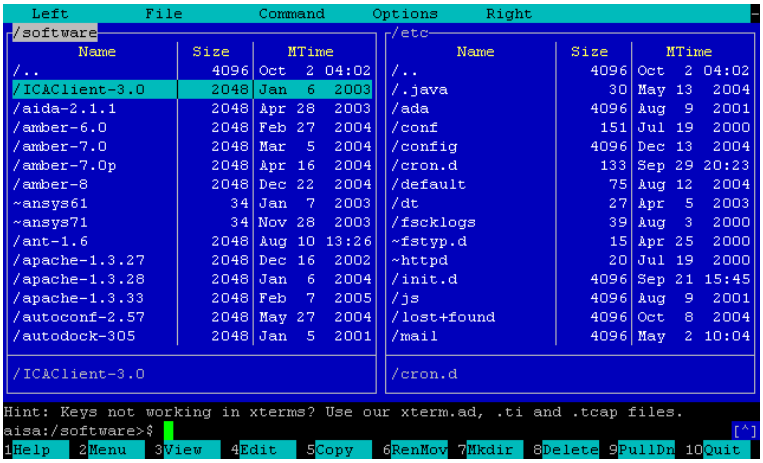
Textová uživatelská rozhraní (Character user interface)

Textové nebo také znakové uživatelské rozhraní [17] pracuje pouze v textovém režimu. V tomto případě bývá obrazovka rozdělena na sloupce a řádky s tím, že do každé pozice je možno zobrazit maximálně jeden znak z předem dané množiny. Pevná hierarchie menu je pro tento režim charakteristická. Při práci s rozhraním je nejprve zapsána akce pomocí znaků a dále je zvolen objekt či jsou vkládána data, pro které se má daná akce provést. Sekvence povelů je v tomto případě předem dána a uživatel je musí znát. Uživatel využívá vstupů z klávesnice pro práci ve znakovém prostředí. Typickým představitelem textového uživatelského rozhraní je příkazový řádek. Toto rozhraní se nejčastěji používá u mainframových systémů, některých operačních systémů či systému MS-DOS.

Grafická uživatelská rozhraní (Graphical user interface)

Grafická uživatelská rozhraní [6] jsou také známa pod zkratkou GUI. Využívájí grafických prvků jako jsou ikony, okna, dialogy, spolu s ovládacími prvky a textem. Uživatel může manipulovat přímo s grafickými prvky zobrazovanými na obrazovce. Na rozdíl od příkazové řádky, je tento způsob interakce se zařízeními jednodušší a více přirozený pro uživatelskou činnost. Výhodou je také to, že není nutné znát specifické příkazy či jazyky.

[1] https://en.wikipedia.org/wiki/Text-based_user_interface#/media/File:Midnight_Commander_(2005)_en.png



Obrázek 2.1: Příklad textového uživatelského rozhraní (Midnight Commander)[1]

Grafická rozhraní jsou většinou první a někdy i jedinou věcí, kterou uživatel z aplikací vidí. Proto velice záleží na prvním dojmu. To může mít vliv na budoucí užívání aplikace, její rozvoj a rozvoj uživatelské komunity.

Obrázek 2.2: GUI operačního systému Ubuntu.[2]

Rysem moderních grafických uživatelských rozhraní je oddělení výstupu aplikace a grafické interpretace [2]. Toto má pozitivní vliv při úpravách aplikace a úpravách GUI. Lze tak jednoduše implementovat více druhů GUI pro široké spektrum zařízení s tím, že na pozadí běží stále stejná aplikace. To vede k úspoře času a financí.

V dnešní době existují programové systémy, které usnadňují tvorbu aplikací s GUI. Umožňují vkládání a editaci prvků rozhraní a také umožňují vývojáři zobrazit pravděpodobný vzhled a rozmístění použitých prvků. Jako příklad mohou sloužit QT Creator, Borland Delphi nebo Visual Studio.

[2] https://www.pcrevue.sk/a/Microsoft-by-sa-mohol-inspirovat--Ubuntu-ponukne-na-jar-moznost-minimalnej-instalacie-bez-vstavanych-aplikacii

Hmotná uživatelská rozhraní (Tangible user interface)

Hmotná uživatelská rozhraní, zkráceně nazývaná zkratkou TUI [21], jsou kombinací vstupů informací o hmotných předmětech s výstupy ze systému či zařízení podporujícího rozšířenou realitu. Virtuální objekty jsou propojeny s objekty fyzickými, se kterými může uživatel manipulovat. To napomáhá lepšímu zážitku při práci s těmito zařízeními a umožňuje také lepší praktické využití v některých profesích.

Jako příklad nám může sloužit vizualizace trojrozměrných objektů. Tato vizualizace je většinou založena na snímání markeru pomocí technik počítačového vidění a zobrazení požadované reprezentace na displeji. Orientace značky pak rozhoduje o orientaci objektu. Jako zařízení může být použit mobil, tablet, ale také průhledové brýle nebo rozšířené zrcadlo (Augmented mirror).

Dalším příkladem může být využití fyzických objektů na interaktivních stolech. Ty se dají použít pro uměleckou tvorbu, ale také pro zábavní účely. Takovéto stoly je možné vidět například v některých technických muzeích. U těchto stolů je také nutné zmínit, že se jedná konkrétně o hmotnou rozšířenou realitu (Tangible Augmented Reality).

Obrázek 2.3: Reactable[3]

2.2 Rozšířená realita

Rozšířená realita [20, p. 2] je definována jako propojení reálného a elektronického světa. Poskytuje jednoduchá okamžitá rozhraní pro elektronické systémy v reálném světě. Dále je na ni možno nahlížet jako na zobrazení reálného světa doplněné o počítačem vytvořené objekty.

Základní otázkou je, jak můžeme uvést rozšířenou realitu do existence. Odpovědí může být specializované zařízení nebo software. Abychom získali fungující zařízení a systém rozšířené reality, potřebujeme minimálně tři komponenty [20, p. 5]. Jedná se o komponentu pro snímání objektů, další pro jejich rozpoznávání v nasnímaných datech a třetí pro vizualizaci. Kromě těchto tří základních komponent se dále také mluví o čtvrté komponentě,

[3] https://reactable.com/live/

a to o prostorovém modelu, který obsahuje informace o reálném i virtuálním světě. Prostorový model slouží pro vhodné spojování obou světů. Nutností je však jejich vzájemná konzistence z hlediska souřadného systému.

Historie

První pokusy o spojení reality a počítačem generovaných informací [20, k. 1] jsou datovány do 60. let 20. století. Za prvního průkopníka technologií virtuální a rozšířené reality může být považován Ivan Sutherland, který v roce 1968 sestrojil první na hlavu uchytitelný displej, takzvaný head mounted display (HMD). Ten však pro svou váhu musel být přichycen také ke stropu místnosti. Displej již měl v sobě zabudované zařízení pro snímání pohybu hlavy a průhledová skla.

Obrázek 2.4: HMD Ivana Sutherlanda[4]

V roce 1975 Myron Krueger založil laboratoř umělé reality Videoplace. Zde se snažil dosáhnout vytvoření umělé reality, která by obklopovala uživatele a dynamicky reagovala na jeho pohyby a akce bez použití brýlí či rukavic. Pro dosažení tohoto cíle využil video kamer, projektorů a speciálně navržených zařízení.

K dalšímu výraznému vývoji technologie rozšířené reality dochází až v 80. a 90. letech 20. století, kdy se tato technologie odděluje jako nezávislý směr výzkumu. Prvního využití v průmyslu se pak tato technologie dočkala v roce 1992 při práci na návrhu kabeláže nového letadla společnosti Boeing. Zde vzniká termín "augmented reality"(AR) tedy rozšířená realita.

V 90. letech dochází s pokusům využít AR ve zdravotnictví, ale také pro výukové trenažery. Koncem dekády vznikají první programové balíčky pro tvorbu aplikací využívajících AR, jako jsou například Wikitude[5] nebo ARToolKit[6].

Mezi roky 1996 až 2001 japonská vláda a společnost Canon Inc. společně investovaly do prácí a výzkumů laboratoře Mixed Reality Systems Laboratory. Největším úspěchem byl první průhledový HMD se stereo koaxiálním videem zvaný COASTAR. Mnoho dalších menších projektů bylo zaměřeno na tvorbu pro zábavní průmysl.

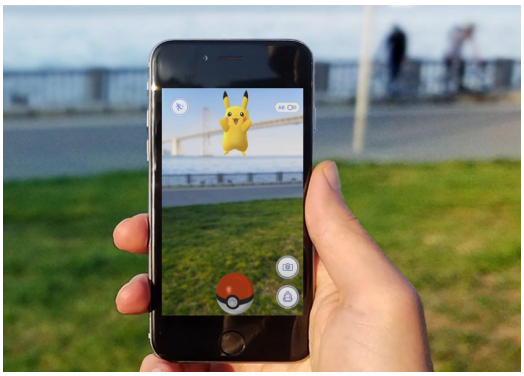
Po roce 2000 díky stále se zvyšujícímu výkonu mobilních zařízení a počítačů začalo být užívání AR více rozšířené. Spolu s tím postoupil i vývoj a výzkum. v roce 2003 představili Wagner a Schmalstieg první ruční zařízení pro rozšířenou realitu. Jednalo se o upravené

[4] http://etsanggarp.blogspot.cz/2016/03/
[5] https://www.wikitude.com/
[6] https://www.hitl.washington.edu/artoolkit/

zařízení “personal digital assistant”, což byl předchůdce dnešních smartphonů. v roce 2005 byla vytvořena první multiplayerová hra pro ruční zařízení zvaná Invisible train, která byla také dostupná pro veřejnost v prezentačních místnostech SIGGRAPH Emerging Technologies.

Velký boom pro rozšířenou realitu [7] přišel v roce 2009. Prvním důvodem byl vznik frameworku ARToolkit, který zjednodušil tvorbu aplikací s využitím rozšířené reality. Druhým důvodem, který upoutal pozornost široké veřejnosti, byl Esquire Magazine, který umístil na svou titulní stránku QR kód. Do obsahu napsal, jaká zařízení a software je třeba pro zjištění, co kód skrývá. Po naskenování kódu se spustilo krátké video s informacemi o rozšířené realitě. Orientace a pozicování videa bylo dáno natočením QR kódu.

Dnes najdeme příklady využití v průmyslu, zdravotnictví či armádě, ale také na osobních zařízeních ve formě her, interaktivních průvodců atd. Dobrým ukazatelem rozvoje může být soupeření společností vyvýjejících brýle pro rozšířenou realitu a také stále větší zapojení těchto technologií do normálního života.



(a) Pokémon GO

(b) Armádní AR od firmy Google

Obrázek 2.5: Příklady užití AR[7]

Specializovaná zařízení umožňující rozšířenou realitu

Pro vizualizaci rozšířené reality existuje mnoho zařízení. Můžeme nalézt množství rozdílů jak po vzhledové, tak funkční stránce. V této části textu se obecně seznámíme s některými z nich.

Brýle

Brýle pro rozšířenou realitu je možné rozdělit na tři typy.

- Prvním typem jsou brýle využívající průhledových displejů, kdy uživatel vidí okolí přímo a virtuální obsah je zobrazen na displeji. Tato technologie se nazývá optical see-through[8]. Příkladem mohou být brýle SeeThru od společnosti Laster Technologies.
- Druhý typ je video see-through, který narozdíl od předchozího typu je uzavřený headset bez průhledových skel. Ten snímá obraz okolí pomocí kamer a po vložení virtuálních objektů do obrazu, jej zobrazuje uživateli.

[7] https://www.vg247.com/2016/08/19/pokemon-go-bans-form/
https://www.armyweb.cz/clanek/rozsirena-realita-pro-vojaky
[8] http://sensics.com/optical-see-through-vs-video-see-through/

- Třetí typ již nevyužívá displeje, ale je založený na promítání obrazu přímo na sítnici uživatele. Příkladem jsou třeba brýle Vaunt od společnosti Intel.

Brýle všech typů často využívají kamery pro snímání hloubky a objektů v okolí. Tyto informace jsou podstatné pro některá využití brýlí.

Interaktivní stoly

Interaktivní stůl je zařízení poskytující možnost interakce uživatele a aplikace. Virtuální výstup může poskytovat pomocí displeje, nebo využíváním promítání na desku stolu. Vstupy mohou pocházet od rozličných vstupních zařízení, jako jsou dotykové plochy, kamery, ale také od objektů umístěných na stole.

(a) SmartEyeGlass[9]

(b) Interactive Smart Table[10]

Obrázek 2.6: Příklad brýlí pro rozšířenou realitu a interaktivního stolu.

2.3 Herní enginy

Herní engine [13] je software pro vytváření počítačových her. Slouží jako funkční jádro počítačové hry. Poskytuje možnosti rychlého a efektivního renderování grafiky, vytváření jednoduché umělé inteligence, jazykových lokalizací hry atd. Výhodou většiny enginů je, že umožňují multiplatformní vývoj, vyřešení nízkoúrovňových problémů za vývojáře hry, nebo možnost si potřebné funkcionality doimplementovat.

Ač herní enginy poskytují mnohé výhody, žádný nebyl využit jako součást aplikace. Přesto v rané fázi vývoje bylo uvažováno o dvou možných enginech, které by byly vhodné pro použití v aplikaci.

- Unity[11] je dnes pravděpodobně nejpoužívanější herní engine. V základní verzi je bezplatný a poskytuje většinu funkcionality. Také existuje verze Pro, která je placená a nabízí pokročilejší funkce. Výhodou je snadný vývoj a specializovaný editor dodávaný spolu s enginem. Součástí je také propracovaná dokumentace, která obsahuje mnoho ukázek a tutoriálů. Jako programovací jazyky lze pro tento engin použít C#, UnityScript, nebo Boo.

[10] http://www.globalmediait-ar.com/sony-lanza-sus-lentes-inteligentes-smarteyeglass/ http://www.itvcomputers.com/interactive-smart-table.php

[11] https://unity3d.com/

- Godot[12] je open-source multiplatformní engine. Licenčně spadá pod licenci MIT. Obsahuje také IDE, jehož hlavní výhodou je, že vývojář nepotřebuje žádný další pomocný software pro vytváření grafiky, úpravu zvuku či dalšího obsahu. Hry lze vytvářet v jazyce C, C++ a v pythonu podobném skriptovacím jazyce GDScript.

2.4 Modelování a simulace

Abychom lépe porozuměli problematice modelování a simulací [19], je nutné si nejprve definovat základní pojmy.

- **Systém** je soubor elementárních částí systému, které mají mezi sebou určité vazby. Systémy můžeme dělit dle různých kritérií. Například podle existence:
  - reálné (existující) systémy - systémy, které jsou reálně analyzovatelné,
  - nereálné (fiktivní, ještě neexistující) systémy – používají se například v počítačových hrách.

  Další dělení může být podle toho, zda mění svůj stav v čase:
  - statické systémy – nemění svůj stav v čase,
  - dynamické systémy – mění stav v čase.

  Dynamické systémy jsou z hlediska použitelnosti zajímavější. Příkladem dynamického systému může být vydávání obědů v jídelně.
- **Model** je napodobenina systému jiným systémem. Důležité je, že model musí napodobovat všechny pro nás podstatné vlastnosti původního systému. Příkladem modelu může být soustava rovnic pro výpočet šíření požáru.
- **Modelování** je proces vytváření modelu na základě našich znalostí o systému. Jelikož kvalita modelu ovlivňuje zásadním způsobem výsledky, je třeba dbát na dosažení co nejvyšší validity modelu.
- **Simulace** je získávání nových vlastností na základě experimentování s modelem. Pro získání potřebných informací a ověření modelu a průběhu simulace, je vhodné simulační experimenty opakovat vícekrát s různými parametry.

Model vždy vychází z podmnožiny našich znalostí, a to na základě zaměření na to, co chceme simulováním zjistit. Modelovat můžeme pouze to, co jsme schopni zjistit, pochopit a popsat. Důvodem pro simulování je, že v reálném světě nemusí být možné požadovaný experiment provádět. Dalšími hledisky, proč simulovat modelem namísto provádění experimentů v reálném světě, jsou například etické zásady, ekonomická náročnost nebo nebezpečnost. Přesto, je-li to možné, je vhodné provádět jak simulace, tak reálné experimenty, aby bylo možné porovnat výsledky.

Alternativou pro simulování je využití analytického řešení modelů pro získání výsledků. Výhodou je, že výsledky těchto řešení bývají přesnější než použití simulace. Nevýhodou je, že tyto metody jsou vhodné pouze pro jednoduše popsatelné systémy. Pro složité systémy většinou není možné jich použít.

[12] https://godotengine.org/

Modelování a následná simulace se využívá v mnoha odvětvích. Příkladem mohou být modely šíření chorob, srážky vesmírných těles, předpověď počasí, ale také počítačové hry. Obecně můžeme říci, že téměř každá počítačová hra má nějaký model, nad kterým je prováděna simulace.

2.5 Počítačové vidění

Počítačové vidění [9] je disciplína snažící se technickými prostředky napodobit lidské vidění. Nejedná se o jedno robustní řešení, ale o soubor technik, jejichž spojení nám umožňuje nahradit lidské vnímání. Jednou z těchto technik jsou také transformace obrazu. Pro řešení problému nezkresleného promítání rozhraní na interakční plochu (např. stůl) je vhodné využít transformaci (homografii) obrazu na rovinný předmět.

Využití počítačového vidění je velice široké. Lze jej použít pro kontrolu slepého úhlu u automobilů, kontrolu kvality výrobků nebo v bezpečnostních systémech. Způsoby a metody počítačového vidění se stále zlepšují, jak po stránce rychlosti, tak po stránce přesnosti.

Detekce žetonů

Detekování žetonů lze provádět na základě mnoha kritérií v závisloti na tom, jaké parametry žeton má. Může se jednat o žeton, který:

- Má na sobě QR kód, který umožňuje jeho detekci a identifikaci. Výhodou je jednoduchost výroby žetonu a také jeho detekce. Pro detekci QR kódu dnes existuje mnoho knihoven, například knihovna Aishack[13]. Nevýhodou je, že pro detekci kódu musí být kód žetonu v obraze čitelný.
- Je odlišné barvy vůči pozadí. Výhodou je jednoduchá výroba. S použitím knihovny OpenCV je možné detekci zjednodušit na několik po sobě jdoucích procesů.
  - Filtrace obrazu na základě barvy žetonu. Výsledkem je dvoubarevný obraz, kde jedna barva nahrazuje barvu žetonů v obraze a druhá vše, co je nehodící se barvy.
  - Využití eroze pro separaci žetonů, jež jsou umístěny blízko u sebe.
  - Vyhledání kontur a jejich umístění v obraze.
- Má charakteristický vzhled. Například se může jednat o tvar nebo vzor. Tato charakteristika se pak používá jako vzor, který je vyhledáván v zachyceném obraze. Tento typ detekce lze také velice snadno implementovat pomocí knihoven.

Homografie

Definice z Learning Opencv [10] říká, že homografie je matematický termín pro mapování bodů jednoho povrchu na body na povrchu jiném. V kontextu počítačového vidění, homografie téměř vždy odkazuje na mapování bodů na dvou rovinách obrazu, které odpovídají stejnému umístění planárních objektů v reálném světě. Zjednodušeně můžeme říci, že jeden objekt představuje předmět ve skutečném světě a druhý pak jeho zobrazení v projekční rovině. Je možné také nalézt homografii pro transformaci objektů mezi rovinami. Mapování lze reprezentovat jedinou ortogonální maticí o rozměrech 3x3, tak jako je ve vzorci 2.1.

[13] urlhttp://aishack.in

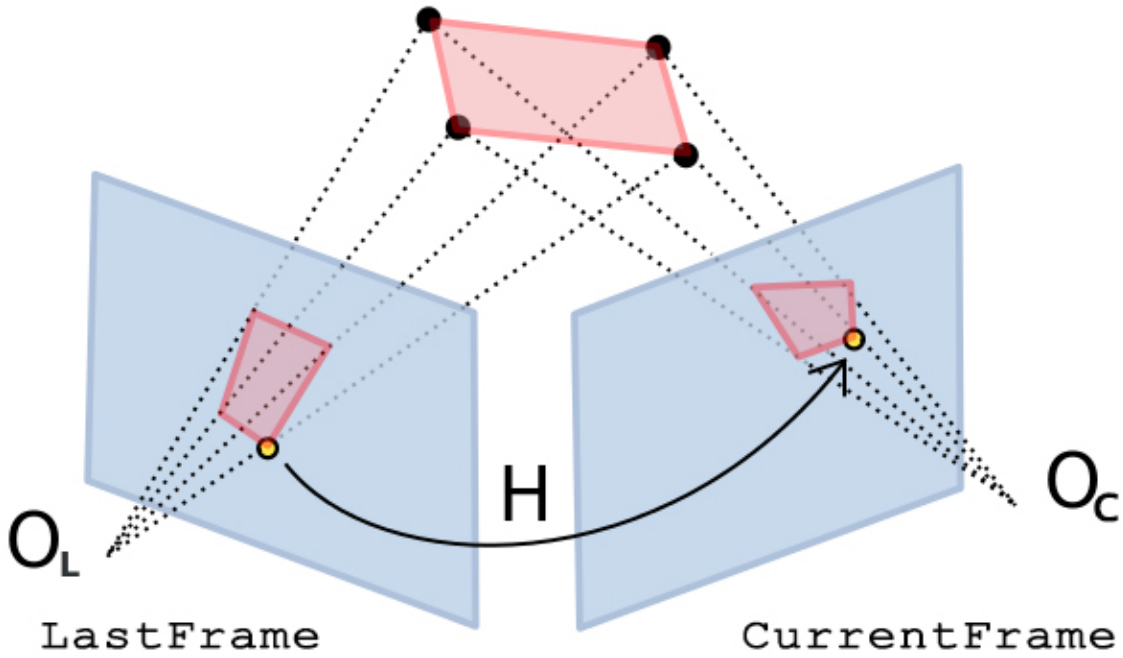
Vzorec vyjádřuje vztah transformační matice a bodů plochy pro získání transformovaných bodů.

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \tag{2.1}$$

Homografii je možné využít pro různé účely, jako jsou kalibrace kamery či jiných optických zařízení, vytváření panoramat z fotografií nebo hledání a transformování objektů v obraze.

Mapování a nalezení matice

Informace pro tuto sekci jsou čerpány z prací E. Dubrofského [12], R. Hartleye a A. Zissermana [14]). Pro nalezení homografie a vypočtení koeficientů matice je třeba alespoň čtyř vzájemně si odpovídajících bodů v obou pohledech. Předpokladem pro jedonoznačnou homografii je, že tři z nalezených bodů nejsou kolineární. Díky těmto bodům je následně možné dopočítat osm koeficientů homografické matice, které vyjadřují vztah mezi objekty v obou pohledech. Pro výpočet není nutné, aby body byly v rozích objektů. Mohou být umístěny různě po objektech, ale vzájemné pozice by měly korespondovat. Pro vyřešení homografické matice se využívá algoritmus DLT (Direct Linear Transform).

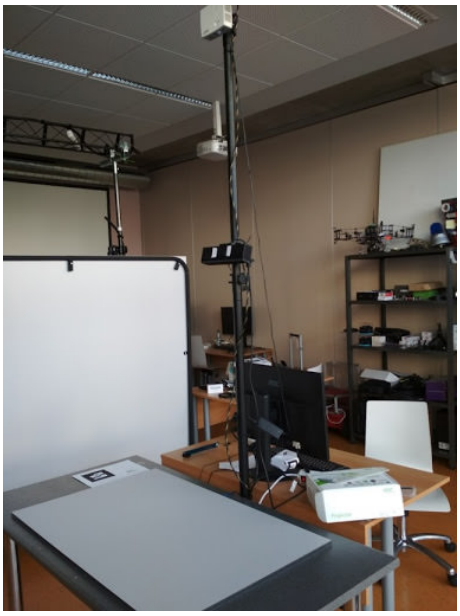


Obrázek 2.7: Obrázkové vyjádření homografie založené na rovnici 2.1[14]

[14] https://www.researchgate.net/figure/Pixel-correspondences-through-homography-between-two-images-With-the-homography-two_fig2_306118034

2.6 ARTable3 (Stůl pro rozšířenou realitu)

ARTable3 je zařízení vytvořené výzkumnou skupinou Robo@FIT[15] na fakultě informačních technologií Vysokého učení technického v Brně. Zařízení umožňuje určité interakce mezi reálným a virtuálním světem. Skládá se z dotykové desky stolu, projektoru a kinectu. Tato sestava umožňuje pomocí kinectu a dotykového povrchu detekovat objekty na desce a jejich přítomnost a působení zavést do systému či aplikací, což uživateli umožňuje ovlivnění systému jiným způsobem než jen využitím myši, klávesnice a jiných běžně dostupných ovladačů. Pomocí projektorů lze do reality promítat data a stavy z právě běžícího systému či aplikace.



Obrázek 2.8: Fotografie interaktivního stolu ARTable3[16]

2.7 OpenCV (Open Source Computer Vision Library)

OpenCV[17] je multiplatformní open-source knihovna pro práci s obrazem se zaměřením na zpracování obrazu v reálném čase. Obsahuje přes 2500 algoritmů pro detekci objektů a pohybu ve videu či obraze. Lze užít také pro kalibraci kamery. Vznikla ve společnosti Intel. Dnes je spravována společností Willow Garage. Je přístupná pod tříbodovou licencí BSD. V knihově jsou implementována rozhraní pro jazyky C++, C, Python, Java a MATLAB. Díky podpoře mnoha jazyků, lze vytvářet aplikace pro velké množství platforem. Podporuje vývoj pro Windows, iOS, Linux, nebo také Android.

Instalace

Vzhledem k tomu, že byla demoaplikace implementována v jazyce C++, bylo nutné při instalaci OpenCV na unixovou distribuci užít cmake s přidanými parametry "D WITH_GTK=ON"

[15] https://github.com/robofit
[16] https://merlin.fit.vutbr.cz/wiki/index.php/RoboLab_ARTable3
[17] https://opencv.org/about.html

a "D WITH_GTK_2_X=ON". V opačném případě aplikace při překladu hlásila chybu s paralelním užíváním GTK2 a GTK3, která znemožnila překlad.

2.8 ROS (Robot Operating System)

Jedná se o framework [3] pro psaní softwaru určeného pro robotická zařízení. Skládá se z nástrojů, knihoven a konvencí, jejichž cílem je zjednodušit vytváření komplexních a robustních programů řídících systémů robotů na celé řadě robotických platforem. Je navržen tak, aby fungoval jako vzájemně komunikující skupina modulů zvaných *nody*. To přináší výhody při vývoji aplikací, neboť je možné snadno práci rozdělit mezi více týmů nebo ji rozdělit na základě specializace.

Node (Uzel)

Uzel[18] je proces, který provádí výpočet. Více uzlů spojených do grafu komunikuje pomocí zasílání zpráv (topic), RPC služeb a parameter serverů. Řídící systém robotů většinou obsahuje mnoho uzlů, kdy každý uzel ovládá a zpracovává data z jiné části hardwaru. Použití uzlů přináší mnoho výhod. Těmi jsou zvýšení odolnosti proti poruchám celého systému, snížení složitosti kódu nebo například možnost, aby každý uzel byl napsán v jiném programovacím jazyce.

2.9 Vývojové nástroje

Zde můžete nálézt stručné popisy použitých vývojových nástrojů a aplikací, spolu se způsobem jejich využití při tvorbě aplikace.

Ubuntu 16.04

Ubuntu je open-source operační systém na bázi Linuxu určený pro stolní počítače, notebooky a servery. Ubuntu poskytuje širokou škálu standardizovaného softwaru, který je taktéž pod licencí open-source. Nové verze s krátkou dobou podpory vycházejí každých šest měsíců. Dále existují verze s dobou podpory 5 let označované zkratkou LTS. V dubnu 2018 vyšlo Ubuntu 18.04 LTS

Operační systém Ubuntu byl podstatný pro tvorbu aplikace z důvodu podpory a snadné údržby operačního systému ROS.

QT

QT[19] je multiplatformní framework používaný pro vytváření aplikací s grafickým uživatelským rozhraním. QT tookit vznikl již v roce 1999 ve společnosti Trolltech. Dnes je spravován společností The Qt Company. Toolkit je možné distribuovat pod licencemi GPL a LGPL. Při splnění specifických podmínek je možné dosáhnout i licence komerční. Knihovna QT byla původně vytvořena pro jazyk C++, ale dnes existuje například také pro jazyky Python, Perl, Pascal, Java či Haskell. Framework obsahuje nejen základ pro tvorbu GUI, ale také podporu SQL, síťových a vícevláknových služeb, případně dalších médií jako je třeba práce se zvukem. Součástí jsou také vývojářská IDE Qt Creator nebo Qt Designer.

[18] http://wiki.ros.org/Nodes
[19] https://www.qt.io

Všechny virtuální komponenty uživatelského rozhraní aplikace jsou implementovány pomocí frameworku QT. Framework je využit také pro síťovou komunikaci při propagaci obrazu pro ROSovský node projektoru.

QT Creator

Je multiplatformní IDE určené především k vývoji aplikací v C++, ale lze v něm tvořit i v jiných jazycích. Poskytuje vizuální debugger určený pro kontrolu implementovaného GUI a jeho formulářů. Součástí je také kontrola syntaxe a správa jednotlivých modulů a zdrojů aplikace.

QT Creator byl využit v rané fázi vývoje aplikace pro vytvoření prvních verzí GUI a funkční logiky. Pozdější přechod na **Catkin** byl náročný, neboť QT Creator využívá pro nastavení překladu svůj vlastní systém **QMake**, zatímco **Catkin** využívá **CMake**.

Catkin

Catkin[20] je oficiální překladový systém v robotickém operačním systému ROS. Je následníkem původně používaného systému rosbuild. Kombinuje makra překladového systému CMake a pythonovské skripty pro funkcionalitu nad rámec CMake. Catkin byl navržen pro umožnění jednodušší distribuce balíků ROSu a jejich přenositelnosti. Postup překladu catkinu je velice podobný překladu pomocí CMake, ale přidává navíc podporu automatického vyhledání balíčků a funkci překladu více vzájemně závislých projektů zároveň.

Zpočátku nebyl Catkin při vývoji používán. Místo něj bylo využíváno překladače v prostředí QT Creator. Přechod na Catkin byl nutný pro práci ROS a implementaci propojení s interaktivním stolem.

CMake

Jedná se o multiplatformní open-source nástroj pro překlad, testování a vytváření balíčků softwaru. [21] Pro nastavení použitých vlastností při překladu je třeba vhodně upravit soubor **CMakeLists.txt**. Cmake pak na základě nastavení vytvoří příslušný makefile a spustí překlad.

[20] http://wiki.ros.org/catkin/conceptual_overview
[21] https://cmake.org

Kapitola 3

Návrh

Tato kapitola se zabývá návrhem systému pro rozšířenou realitu a návrhem hry (demoaplikace), pomocí které bude možné demonstrovat funkcionalitu systému.

3.1 Systém

Systém slouží jako spojnice mezi hrou a softwarem interaktivního stolu. Obsahuje veškerou funkcionalitu pro využívání zdrojů interaktivního stolu pro účely aplikací. Ve své základní formě je přizpůsoben demoaplikaci, ale lze jej velice snadno rozšířit o potřebné části pro podporu dalších funkcionálních a herních konceptů. Systém je nezávislý na demoaplikaci.

3.2 Hra (demoaplikace)

Součástí práce je hra vycházející z konceptu výukových her a aplikací, nicméně není odvozená od žádné známé existující hry. Základem jsou výrobní podniky a cesty, které je spojují. Cílem hráče je umístit žetony na hrací plochu tak, aby byl výsledný generovaný výdělek z podniků co nejvyšší. Hráč se během hraní dozvídá základní informace o výrobních procesech v různých odvětvích. Umísťováním žetonů dochází k aktivaci výrobních podniků a cest. Některé výrobní podniky vyžadují pro své fungování výrobky z jiných podniků. Nevyužité výrobky jsou nakonec přepočítány na utržený zisk. Hru je možné hrát ve více hráčích, a to kompetitivním způsobem. Nejprve odehraje jeden hráč, který když má dojem, že dosáhl výsledku, který již nepřekoná, předá žetony dalšímu hráči. Kolik žetonů bude pro hráče, který rozmísťuje žetony, dostupných je založeno domluvě hráčů před začátkem hry.

Demoaplikace využívá vlastností systému pro interakci s uživatelem.

3.3 Příkladový scénář hry

Zde je jednoduchý příklad hry pro lepší pochopení herních principů vysvětlených v předchozí části.

Po zapnutí aplikace je vygenerována mapa s výrobními podniky a cestami. Po zobrazení mapy na herní plochu může hráč umísťovat žetony na herní plán na jednotlivé podniky a cesty. Po umístění žetonů je analyzován snímek z kamery, na jehož základě je upraven stav hry. Dojde k vypsání statistik jednotlivých podniků a cest na herní plán a vypsání aktuálního výdělku. Hráč může měnit rozložení žetonů na herní ploše, a tím se pokusit

dosáhnout vyššího výdělku. Po každě změně na herním plánu jsou také upraveny informace, jež jsou zobrazovány uživateli.

Předpokládejme, že jsou na herní ploše podniky Vápencový důl (VD) a Cementárna (C), která pro své fungování potřebuje vápenec z VD. Dále předpokládejme, že tyto podniky jsou propojeny cestou. Umístí-li hráč žeton na C, pak celkový výdělek je 0, neboť C nemá suroviny potřebné pro výrobu. Je-li umístěn žeton pouze na VD, pak je výdělek roven přepočtu ceny výrobku na výdělek. Důvodem je, že C ani cesta nejsou aktivní, a proto nemůže být výrobek dále zpracován. Pokud je aktivní jak VD, tak C, ale není aktivní cesta, pak je celkový výdělek stejný jako v předchozím případě. Nakonec, je-li aktivní kromě podniků také cesta, je množství surovin potřebných k výrobě v C posláno po cestě, a zbytek nevyužitých výrobků VD a výrobků C je následně přepočítán na výdělek.

3.4 Vstupy a výstupy systému

Veškeré informace o průběhu a aktivitě na herním stole získává aplikace pomocí kamery. Systém reaguje na změny rozložení žetonů (viz 3.6 Komponenty a jejich funkce) na hrací ploše. Zachycený snímek je pomocí homografie upraven do podoby, která je blízká podobě vnitřní reprezentace. Následné dochází k filtraci na základě barvy žetonů a k vyhledání pozic jednotlivých žetonů na stole (konkrétně uvedeno níže). Na základě výsledků se mění vnitřní reprezentace stavu hry. Ta je promítána skrze projektor na herní stůl. První (počáteční) stav hry je uživateli promítán hned po vygenerování a přípravení herní mapy podle rozložení výrobních podniků a cest.

Detekce žetonů

Detekce probíhá na základě specifické barvy žetonu. Žeton je žlutý a jeho barva není podobná žádné barvě užité v aplikaci. To umožňuje zachycený obraz upravit s použitím filtru hledajícího přítomnost žluté barvy. Na takto upravený obraz je použita transformace nazývaná eroze, využívaná pro jasné oddělení žetonů, které leží blízko sebe. Nakonec probíhá hledání kontur v obraze. Výsledkem hledání je seznam obsahující body, jež jsou vyjádřením přibližných středů jednotlivých žetonů. Detekce probíhá jednou za vteřinu. Důvodem je snížení množství zpracovávaného obrazu a také fakt, že umístění žetonů hráčem oproti průběhu hry, neprobíhá v diskrétním čase.

Přiřazování žetonů k objektům

Výše zmíněný seznam bodů je využit při přiřazování žetonů objektům. To probíhá ve dvou krocích:

1. Přiřazení žetonů k výrobním podnikům.
2. Přiřazení žetonů k cestám.

Již z posloupnosti kroků je zřejmé, že prioritu při zpracování mají výrobní podniky. Je to z toho důvodu, že cesta spojující výrobní podniky vede od středu jednoho, ke středu druhého a do určité vzdálenosti výrobní podniky cestu zakrývají. V případě opačného vyhodnocování, by žeton položený na výrobní podnik, mohl být přiřazen cestě a nikoliv podniku.

Přiřazování žetonů výrobním podnikům probíhá nad celým seznamem dříve získaných žetonů. Žetony jsou přiřazovány pomocí porovnávání souřadnic žetonů s umístěním a rozměry podniků. Po skončení jsou použité žetony vymazány ze seznamu.

Do fáze přiřazování žetonů cestám se dostane již zmenšený seznam. Přiřazování cestám probíhá obdobným způsobem, jako u výrobních podniků. Jediným rozdílem jsou cesty vedoucí mezi podniky tak, že cesta není rovnoběžná s osou x ani y, tedy šikmé cesty. Způsob vyhodnocování těchto cest je podrobně rozepsán v kapitole 4, v oddíle věnujícímu se řešení nejzávažnějších problémů, a to konkrétně v části nazvané **Problematika šikmých cest**.

3.5 Stavy hry

Hra probíhá v diskrétním časovém rámci. Znamená to, že vnitřní reprezentace hry je v danou chvíli v určitém stavu a mění své parametry skokově po vygenerování nového stavu. Přechod do nového stavu hry probíhá po detekování změny rozmístění žetonů na herní ploše.

Pro generování nového stavu hry bylo nutné najít vhodný způsob jak generování provést. Jako možné implementační algoritmy pro generování nového stavu byly uvažovány celulární automaty a genetické algoritmy. Jejich výhodou je, že jsou již formálně popsány, nicméně v průběhu vytváření návrhu systému se ukázalo, že by jejich implementace vzhledem k rozsahu této aplikace byla příliš časově a technicky náročná a nebyl by využit jejich plný potenciál, který se spíše hodí pro aplikace většího rozsahu. Pro generování dalšího stavu je proto využit vlastní "přímý"přístup. Tím je myšleno, že získání dalšího stavu nevychází z již formálně popsaných postupů, ale je navrženo a vytvořeno pouze pro použití v této aplikaci. Používaný způsob je zde pouze pro potřeby demoaplikace. V případě implementování jiné hry nebo vzdělávací aplikace, je samozřejmě možné využít libovolného způsobu generování nového stavu, nebo se plně oprostit od diskrétního časového rámce.

3.6 Komponenty a jejich funkce

Jednotlivé komponenty hry lze zařadit do jedné ze tří skupin. Jedná se buď o žeton, výrobní podnik nebo cestu.

- **Žeton** reprezentuje pracovníka. Jeho umístěním na herní plochu dochází k aktivaci výrobního podniku nebo cesty.
- **Výrobní podniky** produkují výrobky na základě toho, kolik je jejich základní produkční kvóta a počtu zaměstnanců (žetonů). S rostoucím počtem zaměstnanců klesá efektivita jejich práce. Překročí-li počet zaměstnanců určitou mez, pak už přidávání dalších nezvýší produkci. Produkce některých podniků je závislá na produktech předchozích podniků, a to dle typu výrobního řetězce.

  Výrobně zavislé podniky jsou označovány jako výrobní strom. Platí, že podnik ve stromu neprodukuje výrobky, pokud některý z předcházejících podniků není aktivní nebo nemá splněny požadavky na vstupní suroviny.
- **Cesta** funguje jako spoj mezi výrobními podniky. Po cestách dochází k transferu produktů, jež jsou určeny pro další výrobní podniky. Pro fungování cest je nutné, aby zde byl umístěn žeton, který cestu aktivuje. Každý žeton reprezentuje zaměstnance určeného pro přepravu, a ten může přepravit pouze omezené množství zboží. Původní koncept předpokládal, že přidáním více žetonů, lze navýšit množství přepraveného

zboží v neomezené míře. Později došlo k rozhodnutí postupovat stejně jako u výrobních podniků. Při překročení určité hraniční hodnoty, již nejde množství přepraveného zboží zvyšovat. Oba tyto koncepty byly později zavrženy na základě výsledků prvního testování s uživateli.

Pro řešení nedostatku přepravní kapacity bylo připraveno několik konceptů.

1. Kapacita cesty rozdělena rovnoměrně.
2. Váhování dle vzdálenosti, jež musí celkově urazit od výchozího do cílového výrobního podniku.
3. Váhování dle množství jednotlivých výrobků, jenž má být po cestě přesunuto. V takovém případě by měl nejvyšší prioritu výrobek, který má nejvíce jednotek přítomných v uzlu cesty.
4. Koncept založený na přísloví "Kdo dřív přijde, ten dřív mele.". Tento koncept byl během raného vývoje implementován.

Při generování nového stavu dochází procházení jednotlivých komponent a jejich revalidaci. Každý výrobní podnik v pořadí využije co nejvíce z kapacity cesty, aby odeslal své výrobky. Na výrobní podniky, které jsou na řadě později, proto nemusí zůstat žádná volná kapacita.

Tento i další dříve jmenované koncepty však byly po prvním testování s uživateli zavrhnuty, neboť se tyto zdály nepřehledné a složité pro pochopení. V konečné podobě proto počet žetonů na cestě reprezentuje počet podniků, které mohou cestu využívat. Je-li kapacita nedostatečná pro všechny podniky, jež chtějí cestu využít, pak přednost na využití cesty mají podniky, které jsou při revalidaci vnitřní reprezentace dříve v pořadí.

V případě, že navazující výrobní podnik neexistuje, není aktivní, nebo k němu neexistuje aktivní cesta s dostačující přepravní kapacitou, jsou veškeré produkty, jež není možné přepravit přepočítány na výdělek. Součet všech jednotlivých výdělků pak dává dohromady celkový výdělek. Celkový výdělek je spočítán na základě ceny daného typu zboží a jeho množství. Cena je definována v poli. Jelikož má každý typ výrobku své pořadové číslo, slouží toto číslo také jako index v tomto poli.

Propojením výrobních podniků pomocí cest vzniká graf. Při generování komponent je využito průchodu grafu pro nalezení výrobních stromů na herním plánu a uložení cest mezi podniky stromu. Tento počáteční průchod snižuje režiji při zpracování změny stavu, neboť jsou dále využívány pouze dříve nalezené trasy a podnikové stromy.

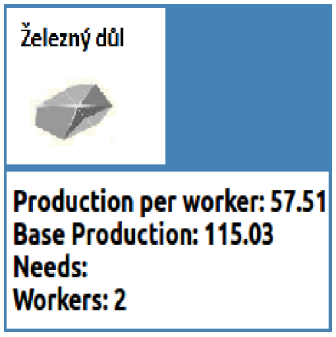
Datový model komponent

Zde jsou popsány vazby jednotlivých komponent z pohledu práce s daty.

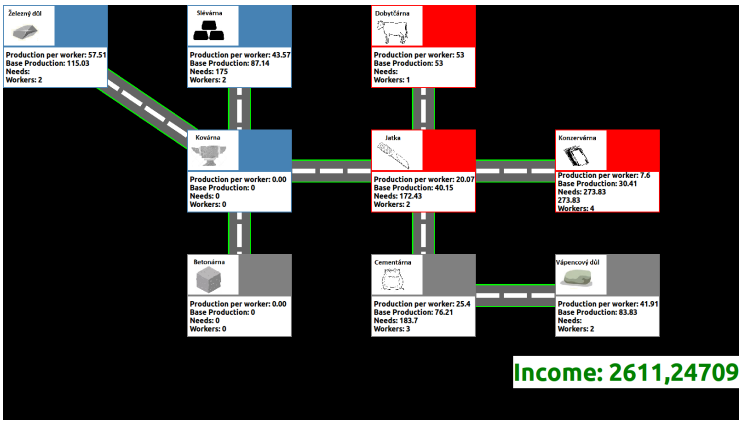
- **Žeton** je reprezentován pouze jako bod umístěný v seznamu všech žetonů.
- **Výrobní podnik** obsahuje své identifikační číslo, výstupní produkt, seznam typů a množství vstupních produktů. Tyto údaje jsou spolu se seznamem cest vedoucích z výrobního podniku, informacemi o předcházejících podnicích a počtem pracovníků nutné pro generování nového stavu hry. Navíc si pamatuje ještě pozici na mapě a velikost v pixelech.
- **Cesta** si uchovává ukazatele na výrobní podniky které spojuje, pozici středu zobrazení podniku na mapě získaného na základě pozice a velikosti podniku a počet pracovníků.

3.7 Návrh grafického rozhraní

Při vývoji grafického rozhraní musíme vzít v úvahu co vše je nutné uživateli zobrazit a jakým způsobem. Podíváme-li se na naši hru, je jasné, že je potřeba zobrazovat hráčům jaký je stav výrobních podniků, cest a celkový výdělek. U výrobních podniků se konkrétně jedná o zobrazení množství vyrobených produktů, jejich typ a množství produktů potřebných pro výrobu. Tyto údaje jsou číselné. U cest bylo nutné vyřešit, jakým způsobem zobrazit kapacitu přepravy tak, aby bylo jasné k jaké cestě patří. Tento problém nebylo nakonec po výsledcích prvního testování třeba řešit. Nakonec bylo nutné vhodně umístit okno pro zobrazení celkového zisku tak, aby nepřekáželo ostatním objektům ve hře. Toho bylo docíleno jeho umístěním v dolní rohové oblasti aplikačního okna.



(a) Detail výrobního podniku



(b) Celkový pohled

Obrázek 3.1: Ukázka prvního konceptu herního rozhraní.

Kapitola 4

Implementace

Tato kapitola má za úkol vysvětlit aplikované metody a postupy implementace programu. Obsahuje stručné popisy nejdůležitějších modulů aplikace a jejich vzájemných propojení. Součástí je také konkrétní popis způsobu propojení ARTable3 a aplikace ve vstupně-výstupní rovině. Implementace vychází z dříve uvedeného návrhu, ale v některých konkrétních částech se může lišit. Důvodem jsou úpravy, které byly založené na výsledcích testování s uživateli. V níže uvedeném textu budou uváděny příklady konkrétních souborů aplikace s odkázáním na soubory s příponou *.h*, kde jsou třídy a funkce deklarovány. Definice jednotlivých funkčních celků jsou pak obsaženy v souborech s příponou *.cpp* stejného jména jako u hlavičkového souboru. Aplikace je založena na principech objektově orientovaného programování, což má pozitivní dopady v oblasti čitelnosti, ale také možnosti jednoduše změnit či přizpůsobit principy demoaplikace k různým potřebám.

4.1 Módy spuštění aplikace

V souvislosti s implementací a využitím programu je třeba upozornit na módy spouštění aplikace. Program je složen ze dvou téměř totožných programů, které sdílejí jádro aplikace, ale nesdílejí vstupně-výstupní části aplikace. První, který je určen k používání na interaktivním stole, je spouštěn standartním způsobem. Druhý je spouštěn přidáním přepínače *pc*. U něho dojde ke spuštění jádra aplikace a zobrazovacího okna, ale pouze na monitoru počítače bez výstupu pro interaktivní stůl. Tento způsob spouštění je určen hlavně pro debugování aplikace, ale může sloužit i jako hra pro počítač. Vstupem jsou kliknutí myši na ikony výrobních podniků. Cesty jsou automaticky nastaveny jako aktivní.

4.2 Okno hry

Po zapnutí programu se zobrazí hlavní okno aplikace. Jedná se o *QWidget*, jehož vlastnosti jsou definovány v souboru **gameEnvironment.h**. Tento soubor obsahuje veškerou základní funkcionalitu grafické reprezentace aplikace. Z důvodu problému kompatibility virtuálních ploch některých operačních systémů a použité vizualizační technologie, je pro hlavní okno nastaveno pevné rozlišení 1920x1080 pixelů[1]s nastavením rozšíření na celou obrazovku. Hlavní okno je po spuštění prázdné. Další komponenty, které se ve hře používají pro zobrazování (**factory.h**, **road.h** a zobrazení zisku), jsou vykresleny, až po vygenerování výchozího stavu hry.

[1] Je-li aplikace spuštěna s přepínačem **pc**, pak je rozlišení pouze 1600x900 pixelů.

Obrázek 4.1: Ukázka herního rozhraní. Snímek je z finální verze aplikace.

Mapa hry

Mapa hry se sestává z výrobních podniků a cest, které jsou deklarované v souborech **factory.h** a **road.h**. Výrobní podnik je *QWidget* a cesta je vykreslena pomocí *QPainter*. Pozice a rozměry jsou spočítány na základě velikosti hlavního okna a pozičního koeficientu určeného prvkem datové struktury *vector* obsahující způsob rozložení jednotlivých výrobních podniků. Vektor rozložení podniků a vektor propojení cestami tvoří pár, který dohromady určuje konečné rozložení herních prvků. Tyto vektory jsou umístěny v souboru **layoutTemplate.h**. Pro podporu více druhů rozložení je databáze všech rozložení implementována jako dvourozměrné pole, kde první sloupec v řádku obsahuje počet podniků a zbytek řádku je jedním typem jejich uspořádání na mapě. To umožňuje jednoduše spravovat a přidávat do databáze další herní uspořádání. Kde je podnik umístěn na herní ploše určuje koeficient z intervalu <0, 1>. Jedná se vektor struktury obsahující dvě čísla. Jedno pro vertikální a druhé pro horizontální osu. Nula odpovídá umístění vlevo nebo nahoře a jednička vpravo nebo dole dle osy, na které se právě pohybujeme.

Herní logika

Logika implementované demoaplikace je umístěna v souboru **demoApp.h**. Třída **DemoApp** je vytvořena v instanci třídy **GameEnvironment**. Při vytvoření se pouze uloží odkaz na otcovskou třídu a vytvoří instanci třídy **layoutTemplate**, která obsahuje počet a způsob rozmístění pro generování podniků.

Generování

Vytváření výrobních podniků a cest je provedeno až po zavolání funkce **generate**. Tato funkce má jako parametry rozměry mapy a určení zda se jedná spuštění s přepínačem *pc*. Parametry rozměrů jsou nutné pro vypočtení a uložení konečné pozice nově vygenerovaného podniku na základě informací ze třídy **layoutTemplate**. Způsob generování zajišťuje, že je-li počet podniků určených rozložením větší než počet typů podniků, pak bude nejprve vygenerován jeden podnik každého typu a zbytek bude určen náhodně. Každému podniku

je také vygenerována základní produkce, ze které je uvnitř třídy **Factory** určeno množství a typ potřebných vstupních surovin. Jelikož je třída **Factory** potomkem třídy **QWidget**, ukládá si pro pozdější zobrazení dle svého typu barvu okrajů, druh ikony a ikony svých produktů a potřebných surovin. Z počátku si také nastaví počet pracovníků na nula.

Po ukončení generování výrobních podniků přichází na řadu generování cest. Jaké podniky cesta spojuje je definováno v **layoutTemplate**. Pro potřeby optimalizace vykreslování a vytváření grafu výrobních podniků si cesta uchovává svoje identifikační číslo, ukazatele na oba koncové podniky a souřadnice jejich středů. Po vygenerování si následně nastaví, stejně jako výrobní podnik, počet pracovníků na nula.

Vytváření vazeb výrobních podniků

Pro rychlejší revalidaci údajů pro vykreslení si každý výrobní podnik uchovává pole datových struktur obsahujících typ a ukazatel na výrobní podnik, který ho předchází ve výrobním stromu. Záměrně je využito pole, neboť předcházejících výrobních podniků může být více, například v případě podniků vyžadujících více druhů vstupních komodit. Samotné hledání těchto předcházejících podniků, pak probíhá ve dvou fázích:

1. Zjištění, zda existují předcházející podniky a kolik jich je. Zde se využívá tří cyklů. Jeden je určen pro podnik, který provádí hledání předcházejících podniků, druhý pro hledané vstupní suroviny a třetí pro listování v seznamu existujících podniků.
   Je-li nalezena shoda, pak je ukazatel na cílový podnik uložen do pomocného pole. Po skončení hledání toto pole obsahuje všechny podniky, které odpovídají požadavkům nyní hledajícího podniku, a je ještě v rámci prvního cyklu posunuto do fáze druhé.

2. Vyhledání a zaznamenání cesty mezi podniky. Pro tuto operaci je využíván jednoduchý rekurzivní algoritmus pro hledání cesty grafem. Jelikož se jedná o algoritmus rekurzivní, provádí se prohledávání do hloubky s využitím systémového zásobníku. Každá cesta obsahuje atribut **visited** typu **boolean**, který je zpočátku nastavený na *false*. Tento atribut slouží k detekci toho, zda již byla cesta při hledání použita. Byla-li použita, je tato proměnná ve stavu *true*. Hledání probíhá na základě hledání shody identifikačního čísla. V případě, že se shodují je zaznamenaná cesta spolu s ukazatelem na podnik uložena do k tomu určené datové struktury a přidána do seznamu předcházejících podniků. Tento seznam je součástí třídy **Factory**, aby si každý podnik uchovával své předchůdce. V případě, že se identifikační čísla neshodují, podívá se algoritmus do seznamu cest, které vedou z nyní navštíveného podniku a vydá se po některé z nich. Takto to pokračuje, dokud není podnik nalezen.

Před další iterací cyklu hledajících podniků se ve všech cestách nastaví atribut **visited** zpět na *false* a vyčistí se všechna používaná pomocná pole pro další iteraci.

Přiřazování žetonů k objektům

Práci s obrazem má na starosti třída **ImageProcessing**, která je deklarována v souboru **imageProcessing.h**. V této třídě je implementace homografie a vyhledávání žetonů v obrazu. Ze třídy **ImageProcessing** obdrží demoaplikace pole typu *vector* obsahující souřadnice nalezených žetonů. Jednotlivé souřadnice žetonů jsou ve vektoru ve formě struktury **winPos** obsahující dva atributy, jeden určený pro souřadnici x a druhý pro souřadnici y. Jak již bylo zmíněno v návrhu, je přiřazování žetonů rozděleno do dvou kroků:

1. Přiřazení žetonů k výrobním podnikům.

2. Přiřazení žetonů k cestám.

Každé přiřazování má svůj cyklus přes *vector* souřadnic. Použité souřadnice jsou ukládány do pomocného *vectoru* a po dokončení cyklu jsou odstraněny ze seznamu.

Vypočtení nového stavu

Když jsou již žetony přidělené jednotlivým podnikům a cestám, je čas na revalidaci jednotlivých komponent a zobrazení. Přepočítávání stavu objektů, je voláno na žádost **GameEnvironment** těsně před překreslením zobrazované herní plochy, a to voláním funkce **getProduction** patřící instancím třídy **Factory**. Výrobní podnik spočítá svůj nový stav a vrátí údaje, které jsou následně zobrazeny v grafickém rozhraní. Jsou-li potřeba údaje z jiného podniku **Factory** pro vypočtení nového stavu, je zkontrolováno, zda tento podnik již vypočítal nový stav. Není-li tomu tak, je tento podnik zpracován přednostně. Při výpočtu se bere v potaz základní produkce stanovená při vygenerování podniku, počet pracovníků (žetonů) a hladina naplnění potřeb pro výrobu. Hladina naplnění potřeb je číslo z intervalu <0,1>. Výpočet probíhá nad množinou potřebných surovin pro výrobu uložených ve *vectoru*, kde jsou uloženy ve struktuře **fTypeNeeds** obsahující typ výrobku a základní potřebný počet. Ten je multiplikován na základě množství pracovníků. U každé suroviny se určí poměr potřeby a naplnění potřeby. Ze všech těchto poměrů se následně vybere nejmenší. Pomocí násobení tímto číslem se upraví celková produkce tak, aby korespondovala s naplněním potřeb.

Cesty při výpočtu nového stavu

Do výpočtu nového stavu se zahrnuje i práce s cestami. Aby mohl podnik zjistit v jakém měřítku jsou naplněny jeho potřeby, je třeba ověřit, že všechny cesty mezi nynějším a zdrojovým podnikem jsou aktivní. Aktivitou je zde myšleno, že je na cestě pro výrobní podnik volný pracovník. Při ověřování se jedná o jednoduchý průchod trasy, která byla zmapována během hledání **vytváření vazeb podniků**. Je-li alespoň jedna z cest na trase neaktivní, nemohou být zdroje z daného podniku využity. Těchto hledání může být provedeno více, v závislosti na počtu předcházejících podniků.

4.3 Propojení s ROS

Veškerá práce s operačním systémem ROS je ve třídě QNode v modulu **qnode.h**. QNode je vytvářen ve třídě GameEnvironment, která je zároveň odkázána jako otcovský prvek. Konstruktor třídy samotné pouze ukládá parametry a odkaz na otcovský prvek. Práce s ROS začíná až po zavolání funkce **QNode::init**. Nejprve dojde k ověření, zda je ROS spuštěn. V případě, že není, je aplikace ukončena. Následuje zavolání skriptu pro kalibraci projektoru. Je vytvořena instance třídy pro zpracování obrazu a začíná odebírání obrazu zachyceného kinectem. Nakonec je nastaven časovač a připojení signálů k funkcím obsahujícím smyčku ROSu a propagaci obrazu na server.

Kalibrace projektoru

Skript kalibrace projektoru je upravenou a specializovanou verzí kalibrace autorů ARTable3. Úpravy spočívají v odstranění kalibrace dotykové plochy a funkcí pro implementaci aplikací v jazyce Python. Samotná kalibrace probíhá v několika krocích.

- Nastavení a příprava kalibračního Scene serveru ze zadaných parametrů
- Promítání šachovnice na stůl a její vyhledání pro získání souřadnic klíčových bodů
- Vypočtení kalibrační matice na základě získaných souřadnic

Tato kalibrace probíhá automaticky v případě, že nebyl ARTable již předtím kalibrován.

Kalibrace pro zpracování obrazových informací

Touto kalibrací je řešen problém získání homografické matice pro pozdější vyhledávání žetonů na stole. Tato kalibrace již vyžaduje přítomnost uživatele. Na stůl jsou promítány čtyři body, kam je nutné umístit herní žetony. Po umístění žetonů je zjištěna jejich pozice ve snímaném obraze a vypočítána transformační homografická matice. Tato matice je pak používána po celou dobu běhu programu.

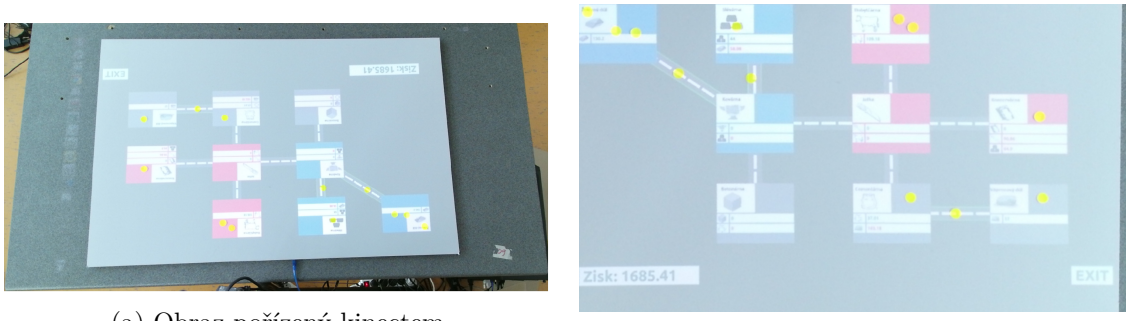
V případě posunutí stolu nebo jiné manipulace se stolem, která mění vzájemnou pozici kamery, projektoru a stolu, je nutné provést obě zmíněné kalibrace znovu.

Obrázek 4.2: Ukázka kalibrace pro homografickou matici.

Zpracování vstupů

Vstupem do programu je obraz zachycený pomocí kamery kinectu. Obraz je transformován tak, aby se co nejvíce blížil původní obrazové reprezentaci, jež je promítána na stůl. Jsou zde využity funkce knihovny OpenCV **findHomography** a **warpPerspective**. Transformační metodou je homografie, jejíž matici jsme získali během kalibrace. Následuje vyhledání žetonů v transformovaném obrazu. Nejprve je aplikován barevný filtr funkcí **inRange**,

která má mezi svými parametry povolené rozmezí barev pixelu ve formátu RGB. Výsledkem je získání oblastí s barevným zastoupením odpovídajícím barvě žetonů. Následuje eroze pro odstranění šumu. Vyhledáváním kontur poté získáme souřadnice jednotlivých žetonů uložených v proměnné typu **vector<vector<Point>**. Konečným procesem je přiřazení nalezených žetonů k pozičně korespondujícím komponentám výrobních podniků a cest. To probíhá průchodem cyklu přes *vector* se souřadnicemi a porovnáváním s pozicí a velikostí podniků a cest.



(a) Obraz pořízený kinectem

(b) Obraz po použití homografie

Obrázek 4.3: Porovnání obrazu před a aplikování homografie.

Zpracování výstupů

Je vyrenderován obraz widgetu **GameEnvironment** a jeho subprvků. Vyrenderovaný obraz se ukládá do proměnné typu **QPixmap**. Pro správně orientované zobrazení na stole je nutné obraz převrátit po vertikální ose. Bohužel pixmapa nepodopuje takovou operaci, a proto je nutné obraz uložit do proměnné typu **QImage**. Takto upravený obraz je nahrán skrze **QBuffer** do **QByteArray**. Důvodem je, že data lze publikovat na serveru pouze v podobě bytového pole. Toto pole je odkázáno do datového proudu (**QDataStream**) a jsou k němu připsány informace nutné pro propagaci obrazu. Jedná se o zařízení a velikost dat. Nakonec jsou data propagována na Scene server, kde si je node spravující zobrazování dat na projektoru převezme a v případě úspěšného přečtení zobrazí.

4.4 Zajímavé a problematické části řešení

Při implementaci nastalo několik situací, jejichž řešení zabralo více času a úsilí. Zde jsou popsány nejvýraznější z nich a způsob jejich řešení použitý v programové části této práce.

Problematika šikmých cest

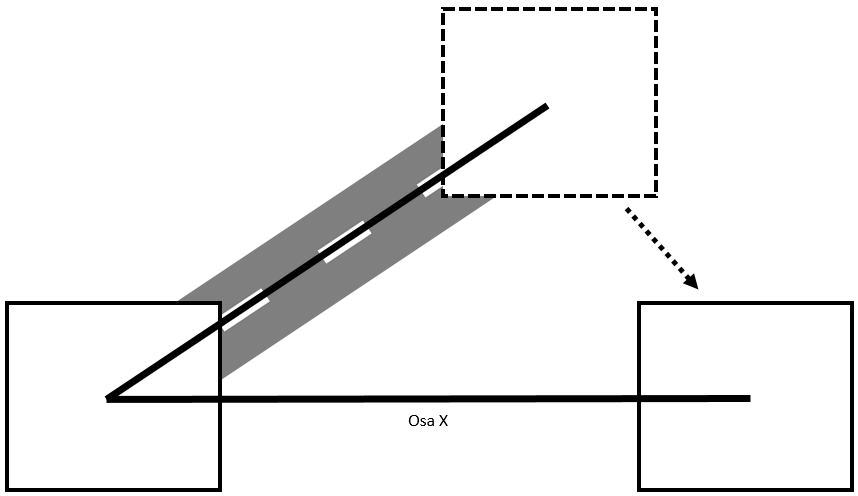
Cesty vedoucí mezi výrobními podniky, jejichž vzájemné pozice nemají shodnou ani jednu ze souřadnic x a y, lze nazvat jako šikmé. Jelikož jsou cesty charakterizovány jako spojnice výrobních podniků, můžeme si je představit jako úsečku spojující středy výrobních podniků. Středy byly zvoleny z důvodu lepšího vzhledu v grafické reprezentaci. Nejprve se podívejme na příklad cest vedoucích vodorovně a svisle. Díky již dříve uvedené propojenosti a dostřednosti, je jednoduché určit, zda je žeton na cestě umístěn nebo zda je umístěn mimo cestu. Máme koncové body úsečky AB a známe šířku cesty. Z toho můžeme zjistit protilehlé body obdelníku, který ohraničuje celou cestu, jednoduchým výpočem:

$$\begin{aligned} O_{1x} &= A_x \\ O_{1y} &= A_y + W \\ O_{2x} &= B_x \\ O_{2y} &= B_y + W \end{aligned} \tag{4.1}$$

Vyjádření zobrazené ve vzorci 4.1 platí pro cestu vedoucí vertikálně. Pro horizontální cestu je třeba přepočítat souřadnici x. O značí bod obdelníku, A a B jsou středy výrobních podniků a W je šířka cesty.

Pak už zbývá pouze otázka, zda bod, kde byl žeton detekován, leží v oblasti mezi těmito body.

Toto velice jednoduché řešení pro vertikální a horizontální cesty však nelze aplikovat na cesty šikmé, neboť by žetony buď nikdy nebyly v rozsahu mezi vypočtenými body nebo by rozsah byl příliš velký a zahrnoval i prostory mimo cestu. Proto bylo nutné vymyslet buď speciální způsob určení pro to, zda leží žeton na šikmé cestě nebo transformaci, která nám šikmou cestu převede do paralelního stavu s vertikální nebo horizovnální osou.



Obrázek 4.4: Princip transformace.

Jako řešení byla zvolena možnost transformace. Výhodou bylo, že bylo možné následně použít algoritmus již implementovaný pro dříve jmenované typy cest. Transformace samotná využívá vektorové algebry. Stanovíme-li si u jednoho podniku (pro jednoduchost jej nazveme P), že jeho střed leží na ose x, pak si můžeme na stejné ose zvolit libovolný bod a vytvořit vektor definující osu x. Následně vypočteme vektor ze středů obou výrobních podniků. Úhel (viz. 5.5), jež svírají tyto vektory, je zároveň úhlem o který musíme otočit úsečku cesty a také úsečku mezi výrobním podnikem P a žetonem. Po otočení pak způsobem popsaným dříve pro vertikální cesty určíme, zda žeton leží na cestě.

```
dot = va.x*vb.x + va.y*vb.y;      // dot product between [x1, y1] and [x2, y2]
det = va.x*vb.y - va.y*vb.x;      // determinant
angle = (-1)*atan2(det, dot);
```

Obrázek 4.5: Ukázka výpočtu úhlu pro otočení cesty.

Křížení cest

Dalším problémem, který se může v souvisloti s cestami a umísťováním žetonů na ně objevit, je jejich křížení. Tato problematika se týká také cest, které dříve neklasifikuji jako šikmé. Může nastat situace, kdy je rozložení výrobních podniků a cest takové, že se cesty v nějakém místě kříží. Je nutné zabývat se otázkou, co se stane, když na toto křížení umístíme žeton. Které cestě bude příslušet pracovník symbolizovaný žetonem? Existuje více možností řešení:

- Ignorování žetonu
- Přiřazení žetonu jedné z cest
- Přiřazení žetonu oběma cestám

V systému je tento problém řešen tak, že pracovník přísluší cestě, která je dříve v pořadí při revalidaci stavu komponent.

Propagace obrazu na Scene server

Ač je již tato problematika zjednodušeně popsána v části **Zpracování výstupů** bylo by vhodné, se k tomuto problému vrátit podrobněji. Vzhledem k tomu, že byla aplikace implementována v jazyce *C++*, ale interaktivní stůl je primárně psán v jazyce *Python*, bylo třeba vyřešit jakým způsobem se liší implementace vytváření *Scene serveru* a propagování obrazu na něj. Toto hledání bylo založeno na referenční aplikaci psané v jazyce *Python* přítomné na ARTable3. Je z balíčku **art_projected_gui** a nazývá se **example_gui.py**. Vytvoření serveru se nijak neliší. Jak je psáno výše v části **Zpracování výstupů**, pro server je využíván *QTcpServer*. Jelikož je v obou aplikacích využito grafického frameworku QT, neliší se příliš ani ukládání obrazu do datového streamu. Hlavní a velice podstatný rozdíl přichází až při zapisování velikosti dat obsažených v datovém streamu.

```
out.device().seek(0)
out.writeUInt32(block.size() - 4)
```

Obrázek 4.6: Ukázka problematického místa v Pythonu

```
out.device()->seek(0);
out << (quint32)(buffer.size() - 4);
```

Obrázek 4.7: Ukázka problematického místa v C++

Problémem je, že jazyk C++ pro práci s datovým streamem obsahuje funkce s podobným názvem jako funkce v jazyce Python. Konkrétně se jedná o funkce **writeBytes** a **writeRawData** avšak jejich funkcionalita je odlišná a nemohou být v tomto případě použity. Jediným funkčním ekvivalentem je použití « a přetypování do požadovaného datového typu (viz obrázek 4.7 a 4.8).

Kapitola 5

Testování

Cílem testování je vyhodnocení použitelnosti aplikace a uživatelské spokojenosti. Údaje pro další vývoj či hodnocení již hotové aplikace získáváme pomocí zpětné vazby testujících. Pro testování byly využity principy známé pod zkratkou **FURPS**. [18]

- Functionality (funkčnost) - Jak je funkční, zabezpečená, portabilní. Zda obsahuje aplikace to, co uživatel opravdu potřebuje.
- Usability (použitelnost) - Jak je aplikace použitelná z pohledu uživatele, její estetika a zda je dokumentace k aplikaci dostačující.
- Reliability (spolehlivost) - Jaká je přesnost výsledků, dostupnost, stabilita, střední doba mezi selháními, apod.
- Performance (výkon) - Jaká je rychlost, responzivita, využívání zdrojů, vytěžování sítě, atd.
- Supportability (udržovatelnost) - Jaké jsou možnosti nápravy nedostatků, vylepšování, adaptace, testování a flexibility.

Spojením výstupů subtestů zaměřených na jednotlivé metriky vzniká testovací sada. Výstupem aplikace testovací sady jsou data, která po analýze odhalí největší slabiny aplikace. Pro efektivní testování je třeba stanovit si plán [1], jakým bude testování probíhat:

- **Analýza cílů testování** - získání souboru cílů, které jsou podstatné pro další vývoj aplikace a její funkcionalitu
- **Způsob testování** - popis testovací metody a způsobu jejího nasazení, definice rolí v testovací metodě
- **Cílová skupina** - základní popis typu uživatele, kterému je aplikace určena a způsob výběru jedinců k testování
- **Způsob vyhodnocení testů** - Vyhodnocení dat získaných testy, určení silných a slabých stránek nynější implementace

5.1 Analýza cílů testování

Jelikož je aplikace definována jako systém pro rozšířenou realitu a demoaplikace, je nutno definovat cíle pro každou část. Společně tyto cíle tvoří množinu všech cílů nutných k testování.

Cíle testování systému pro rozšířenou realitu

Cíle pro tuto část aplikace jsou spojeny hlavně s prací se samotným systémem a jeho užíváním. Část tohoto testování je nutné provádět na alespoň trochu znalých uživatelých. Konkrétně se jedná o:

- Náročnost zavádění a obsluhy systému uživatelem na interaktivním stole
- Funkcionální chybovost

Cíle testování demoaplikace

Demoaplikace na rozdíl od sytému již obsahuje cíle testování zaměřené na grafickou stránku aplikace a

- Náročnost pochopení herních cílů
- Orientaci v herním prostoru a vizuálních prvcích
- Fungování herních prvků
- Propojení demoaplikace se systémem

Cílová skupina

Demoaplikace byla navržena jako jednoduchá výuková hra, která má za úkol v hráčích probudit soutěživost (snaha dosáhnout co největšího výdělku) a zároveň je poučit o fungování výroby běžně dostupných komodit. Hra není zaměřená na žádnou věkovou kategorii, ale pro lepší pochopení a zvládnutí ovládání, bylo třeba zjednodušit některé prvky. Předpokladem na minimální věk hráče je 10 let[1], a to z důvodu znalostí nabytých na základní škole v oboru matematika. S touto skupinou lze provádět spolu s testy zaměřenými na rozhraní a zábavnost demoaplikace, také základní testy prvků ovládání systému v oblasti reagování systému na vstupy. Důraz testování je kladen zejména na již dříve jmenované pochopení hry a ovládání. Ideálním příkladem uživatele, patřícího do této skupiny, je dítě ve věku 12 let, což odpovídá přibližně šesté třídě základní školy.

Skupina subjektů starších 20 let je pak vhodná pro testování logických prvků a systému samotného. Pro zjištění náročnosti obsluhy uživatelem, je zapotřebí mít ve skupině zástupce uživatelů znalých užívání technologie interaktivního stolu, ale také uživatelů neznalých. Tyto dvě skupiny uživatelů nemusí být tak obsáhlé jako skupina určená pro testování samotné demoaplikace.

[1] Je možný i nižší věk, v závislosti na mentálním rozvoji.

5.2 1. testování

V prvním testování bylo hlavním cílem testování uživatelského rozhraní demoaplikace a zábavnosti. V této fázi ještě nebyla demoaplikace zprovozněna na interaktivním stole a testování probíhalo na počítači. Rozhraní bylo zobrazováno na obrazovce a interakce byly prováděny myší. Vzhled rozhraní odpovídal vzhledu v obrázku 3.1 s tím rozdílem, že u každé cesty byla textová bublina, která zobrazovala dostupnou a využitou přepravní kapacitu cesty a počet přidělených pracovníků.

Testovací skupina

Skupina testujících byla složena celkově z 24 osob. Věkové rozvrstvení bylo 14 testujících ve věkové skupině 10-17 let, 4 ve věku 18-25 let, 3 ve věku 26-40 let a zbývající 3 ve věku 41-99 let. Největší důraz byl kladen na skupinu nejmladších testujících.

Použité metody

Na testovací skupinu byly využity dvě odlišné metody testování. Pro věkový rozsah 10-17 let byla použita metoda moderovaného testování použitelnosti [5]. Tato metoda se zakládá na přímé interakci testovaného s aplikací i s vedoucím testování. Testující se může kdykoliv dotázat na informace ohledně fungování či možností aplikace. Moderátor slouží jako pozorovatel a v případě potřeby poskytovatel infomací hráči. Výhodou také je větší kontrola nad kroky, jež subjekt provádí k dosažení cíle.

U věkových kategorií 18-99 let byla využita metoda testování použitelnosti bez moderátora [4]. Důvodem byl předpoklad větší samostatnosti respondentů této věkové kategorie.

Průběh testování

Nejprve je třeba říci, že nemladší kategorie testovala aplikaci na jednom místě a další testující začínal bezprostředně potom, co testování ukončil předchozí testující. Starší věkové skupiny neměly jednotné místo ani čas a byla menší šance vzájemného ovlivnění mezi testujícími.

Testování probíhalo v několika fázích:

1. Představení aplikace a cílů testování.
2. Vysvětlení způsobu interakce s aplikací. (Pouze u mladších testujících)
3. Zadání úloh testujícímu a následná možnost volného užívání aplikace.
4. Pozorování interakcí uživatele s aplikací a pořizování zápisu kroků testujícího.
5. Po ukončení interakce rozhovor o aplikaci, jejím ovládání, pochopení a hodnocení.

Způsob hodnocení

Během hodnocení a vyhodnocování výsledků testování bylo nutné stanovit si způsob hodnocení založený na již dříve zmíněných principech. V našem případě se jedná konkrétně o:

- Čas nutný pro pochopení hry a jejího ovládání

- Úspěšnost vypracování úloh a čas nutný pro jejich splnění.
- Reakce testujícího na nečekané události.
- V případě dobrovolného dalšího testování, po ukončení zadaných úloh, jakým způsobem testující postupuje a proč.
- Zhodnocení kladů a záporů aplikace.

První dva body byly zaměřené na vypracování úloh a očekávaným výstupem bylo rozložení podobné jako je tomu u gaussova diagramu. Reakcemi na nečekané události je myšlen hlavně způsob vyřešení problému v případě, že uživatel narazí na chybu aplikace. Postup při dalším volném testování je důležitý pro odhalení možností a případně poruch, které testování pomocí úloh nemuselo plně pokrýt. Poslední bodem je zhodnocení uživatele. Zde je nutné mít na vědomí, že se jedná o subjektivní hodnocení.

5.3 Vyhodnocení výsledků

Hlavně díky většímu počtu testujících z nejnižší věkové kategorie, bylo možné odhalit problémy uživatelského rozhraní a jeho vlivu na zábavnost a pochopení cíle hry. Starší věkové kategorie, mimo již zmíněné problémy, upozorňovali na nepříliš dobře provedený matematický model aplikace. Testování se dá jako celek považovat za úspěšné, neboť splnilo cíle stanovené pro toto testování.

Chyby uživatelského rozhraní

Největší obtíže dělala interpretace informací o cestách, ale ani zobrazení informací o výrobních podnicích neskončilo bez výhrad. Jednalo se hlavně o zobrazování velkého množství informací bez řádného vysvětlení. Vysvětlení bylo uváděno textově a v některých případech mohlo splývat se zobrazovanými informacemi.

Funkční chyby

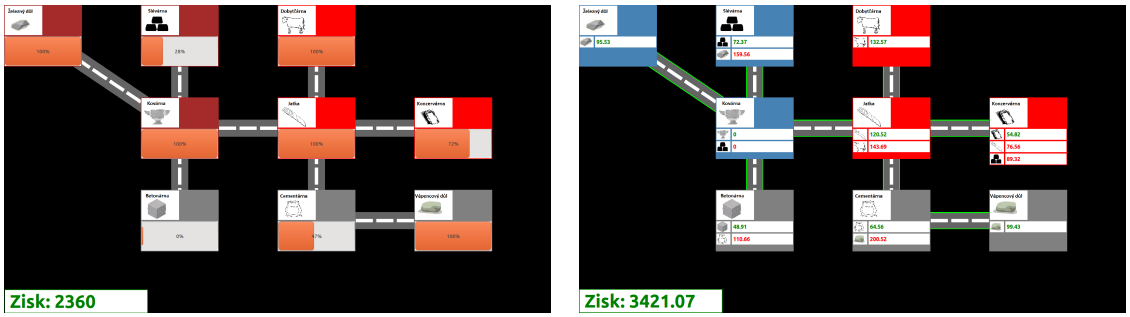
Bylo objeveno několik funkčních chyb, které značně ovlivnily výsledky testování. Jednalo se o nedokonalý matematický model. Nedokonalý generátor rozložení výrobních podniků a cest na mapě hry. Některé výrobní podniky se přibližně z 5% kryly a křížení cest či vedení cest pod skupinou podniků v některých případech znemožňovalo cestu aktivovat, nebo ji vůbec spatřit.

Návrhy vylepšení a rozšíření programu

Testování ukázalo, že je nutné přepracování uživatelského rozhraní. Řešením by mohlo být snížení počtu informací u všech objektů. Nejvíce problémů testujícím dělali informace o cestách, proto by mohlo být vhodné zkusit navrhnout fungování cest tak, aby nebylo nutné některé nebo všechny informace zobrazovat. Také by mohlo pomoci, kdyby jednotlivé druhy informací byly uvozeny symbolem nebo obrázkem. Spolu s vyřešením problému s matematickým modelem aplikace, je možné také upravit fungování některých částí vnitřní interpretace pro efektivnější fungování aplikace. Nynější struktura zavedení modelu totiž má problém s velkým počtem průchodů přes vektory výrobních podniků a cest. Generátor rozložení objektů není příliš podstatnou komponentou, a proto by bylo možné jej odstranit a nahradit modulem obsahujícím již předem připravená rozložení.

5.4 2. testování

Druhé testování bylo více zaměřeno na fungování vytvořeného systému na interaktivním stole. Zvláště na spuštění aplikace a kalibraci pro získání matice pro homografii, z pohledu uživatelské přívětivosti. Součástí bylo také ovládání demoaplikace pomocí prvků implementovaných v systému a náročnost jejich použití. Testování ověřovalo také vhodnost nových návrhů uživatelských rozhraní. Návrhy se týkají především způsobu zobrazení informací o výrobních podnicích a aktivitě cest.



(a) První návrh nového rozhraní.(Progress bar) (b) Druhý návrh nového rozhraní.(Tabulky)

Obrázek 5.1: Porovnání návrhů nového rozhraní.

Testovací skupina

Testování se účastnilo 5 subjektů patřících do věkové kategorie 18-25 let. Dva testující již měli s prací na interaktivím stole ARTable3 zkušenosti, jak uživatelské tak vývojářské. Zbytek testujících nemělo zkušenosti s interaktivním stolem a jeho náležitostmi. Měli však zkušenosti s vývojem a zaváděním softwaru na jiných platformách, což bylo při testování patrné.

Použité metody a speciální faktory testování

Při testování byla použita jediná metoda. Jednalo se o testování použitelnosti. Tato metoda již byla použita při prvním testování, kde se osvědčila. Součástí testování byl stručný návod, který obsahoval způsob, jak je možné aplikaci nasadit na interaktivní stůl a jak ji spustit.

Průběh testování

Před samotným testováním bylo nutné testujícím, kteří dosud neznali ARTable3 vysvětlit základy jeho fungování, aby mohl testující zkusit splnit všechny testovací úlohy.

Testování probíhalo v několika fázích:

1. Představení aplikace a cílů testování.
2. Vysvětlení způsobu interakce s aplikací.
3. Zadání úloh testujícímu, které zahrnovalo také nasazení aplikace na interaktivní stůl a její spuštění.
4. Pozorování interakcí uživatele s aplikací a pořizování zápisu kroků testujícího.

5. Po ukončení interakce rozhovor o aplikaci, jejím ovládání a pochopení. Jelikož se jednalo i o znalé testující byla možnost návrhu vyřešení konkrétního problému.

Způsob hodnocení

Jako u předchozího testování bylo nutné si připravit parametry, podle kterých bude hodnocení a následné vyhodnocování zpracováno. Pro druhé testování byly některé položky upraveny vzhledem ke zkušenostem testujících a také cílům testování.

- Čas nutný pro nasazení aplikace, její spuštění a kalibraci.
- Čas potřebný pro pochopení hry a jejího ovládání.
- Úspěšnost vypracování úloh.
- Reakce testujícího na nečekané události.
- V případě dalšího dobrovolného testování po ukončení úloh, jakým způsobem testující postupuje a proč.
- Zhodnocení kladů a záporů aplikace.

Narozdíl od prvního testování je první bod zaměřen pouze na přípravu hry. Další dva body jsou stejné jako u předchozího testování. Výhodou oproti prvnímu testování v oblasti reakce na nečekané události byl fakt, že někteří testující byli obeznámeni s problematikou, a proto mohli lépe vystihnout původ chyb. Posledním bodem je opět subjektivní zhodnocení uživatele. V tomto případě užitečné pro výběr stylu uživatelského rozhraní.

5.5 Vyhodnocení výsledků

Testování, ač probíhalo na malém vzorku testujících, je možné označit za úspěšné. Odhalilo několik chyb ze sledovaných kategorií. Porovnání obou nových možností uživatelských rozhraní přineslo neočekávané výsledky v oblasti zvýraznění aktivních cest.

Chyby uživatelského rozhraní

Nejvíce zmiňovaným problémem byla absence možnosti ukončit hru přímo na desce stolu. Dosavadní zvýraznění aktivní cesty bylo uznáno jako nedostatečné, neboť promítaní na desku interaktivního stolu způsobilo nízkou viditelnost zvýraznění. Z možností zobrazení informací o výrobních podnicích vyšlo najevo, že první návrh je nepřehledný.

Funkční chyby

Hlavní funkční chyba nastala při snaze vytvoření homografické matice při umístění více než čtyř žetonů na stole. Tato chyba způsobovala vytvoření nesprávné matice, a tím zabraňovala hledání žetonů. Jednalo se o chybu v podmínce při hledání žetonů pro kalibraci. Chyba byla hned po nalezení opravena, aby nekomplikovala další testování. Další chybou bylo, že při provádění kalibrace nebylo zřejmé, co má testující udělat, aby ke kalibraci došlo.

Návrhy vylepšení a rozšíření programu

Nejprve je nutné zmínit úpravy, jež byly zavedeny během nebo přímo po testování. Na základě zpětné vazby byla během testování přidána možnost vypnout hru pomocí umístění žetonu na objekt ve spodní části stolu, který je pro tuto akci určen. Zpětná vazba rovněž umožnila konečný výběr uživatelského rozhraní pro finální verzi aplikace. Byla vybrána druhá možnost(viz obrázek 5.1) z rozhraní testovaných v tomto běhu testování. Také se ukázala výhoda využití tohoto typu rozhraní v případě, že by podnik produkoval více druhů zboží.

Do konečné verze by bylo vhodné přidat pokyny pro uživatele při kalibraci, kdy aplikace získává data pro spočtení homografické matice. Pokyny poslouží uživateli k lepší orientaci, jak postupovat, aby kalibrace započala a úspěšně proběhla. Mohlo by se například jednat o výraznější vyznačení bodů pro položení žetonů a také text, který by uživatele instruoval.

Kapitola 6

Závěr

Cílem bakalářské práce bylo vytvořit systém, který by bylo možné použít pro tvorbu deskových her a výukových aplikací využívajících interaktivní stůl.

Pro dosažení cíle bylo nutné důkladně nastudovat technologické teorie zabývající se rozšířenou realitou a tvorbou grafických rozhraní. Důležitou součástí bylo objevení implementačních možností programovacích jazyků a knihoven v porovnání s možnostmi interaktivního stolu ARTable3. S tím také souvisí analýza dostupných technologií, které jsou vhodné pro použití v aplikaci.

Na začátku byl vytvořen předběžný návrh demoaplikace a systému pro práci s interaktivním stolem. Tento návrh v sobě zahrnoval výběr vhodného programovacího jazyka, technologií a vývojových nástrojů. Vybrané použité nástroje bylo nutné na základě zkušeností s implementací v několika případech nahradit jinými, pro tento účel vhodnějšími, např. QT Creator vs. Catkin. Kromě vyřešení uvedených technologických otázek, byla v této části práce navržena samotná hra, jež se stala podkladem pro vývoj demoaplikace. V rámci návrhu hry bylo třeba vytvořit datový model, způsob jeho komunikace s uživatelem skrze vstupní a výstupní části systému a připravit uživatelsky přívětivé grafické rozhraní.

Značná část práce se věnuje implementaci návrhu a způsobu, jakým bylo dosaženo nynějšího stavu aplikace. Během implementace byla vytvořena aplikace, která je spustitelná a využitelná jak na interaktivním stole, tak na běžném počítači. Verze pro počítač byla určena převážně pro implementaci herní logiky a funkcionality hry. Po dokončení demoaplikace bylo možné ji napojit na vytvořený systém určený pro komunikaci s interaktivním stolem, což umožnilo následné dokončení celé práce. Výsledkem implementace je funkční aplikace, která obsahuje systém pro práci s interaktivním stolem založeným na snímání objektů pomocí kinectu a samotnou hru, která systém využívá k demonstrování a testování výsledků práce. V průběhu implementace bylo třeba vyřešit také několik problémů, a to získání matice pro homografii skrze uživatelskou kalibraci, problematiku šikmých cest, křížení cest a v neposlední řadě propagaci obrazu na scene server. Všechny tyto problémy byly zcela vyřešeny.

Poslední část práce obsahuje metodiku použitou pro testování, stanovení postupů a cílů testování a jeho rozbor. V průběhu implementace systému a demoaplikace byla provedena dvě testování. První bylo zaměřeno převážně na uživatelské rozhraní demoaplikace a druhé na fungování systému na interaktivním stole. Na základě testování byly odstraněny nalezené funkční chyby a upraveno uživatelské rozhraní do současné podoby. Testování zároveň přineslo další návrhy k rozšíření vytvořeného programu v budoucnu.

Cíl práce, tedy vytvoření fungujícího systému a demoaplikace byl naplněn, nicméně v systému je velký prostor pro celkový rozvoj funkcionality. Vzhledem k tomu, že byl systém

přizpůsoben demoaplikaci, bylo pro práci s interaktivním stolem využito pouze projektoru a kinectu. Naopak nebylo využíváno, ani nijak zahrnuto, ovládání dotykové desky interaktivního stolu. Tato problematika by mohla být předmětem dalšího vývoje systému. Také by bylo možné pokusit se přidat podporu snímání a identifikace gest ruky. Tyto způsoby ovládání by mohly být dobře využity nejen u klasických deskových her, ale například také karetních her převedených do virtuální reality. Pro některé hry a aplikace by mohla být užitečná implementace rozlišování objektů na základě hloubky či výšky objektu v obraze. Jmenované návrhy se zatím týkaly pouze nových možností vstupů systému. Další rozšíření by mohla vést k lepší podpoře hry více hráčů, a to nejen na jednom stole, ale také při používání většího počtu interaktivních stolů. Příkladem by mohl být modul, který by zastřešoval komunikaci mezi interaktivními stoly. Tak by mohli hráči hrát, i když by nebyli přítomni u jednoho stolu.

Možnosti interaktivních stolů a rozšířené reality obecně mají velký potenciál a díky stále probíhajícímu vývoji, by mohly vzniknout nové možnosti jejich využití.

Literatura

[1] *Improving the User Experience.* [online; navštíveno 27.4.2018].
URL https://www.usability.gov/

[2] *Návrh uživatelského rozhraní webové aplikace.* [Online; navštíveno 29.4.2018].
URL http://gml.vse.cz/data/oppa-webdesign/ui.html

[3] *ROS.* [Online, navštíveno 5.2.2018].
URL http://www.ros.org/about-ros

[4] *Running a Usability Test.* [online; navštíveno 20.4.2018].
URL https://www.usability.gov/how-to-and-tools/methods/running-usability-tests.html

[5] *Using moderated usability testing.* [online; navštíveno 20.4.2018].
URL https://www.gov.uk/service-manual/user-research/using-moderated-usability-testing

[6] *Typy uživatelských rozhraní a jejich specifika.* [Online; navštíveno 29.4.2018], 2013.
URL https://wikisofia.cz/wiki/Typy_u%C5%BEivatelsk%C3%BDch_rozhran%C3%AD_a_jejich_specifika/old

[7] *Brief History of Augmented Reality.* [Online; navštíveno 24.4.2018], 2017.
URL https://www.igreet.co/brief-history-of-augmented-reality/

[8] Aukstakalnis, S.: *Practical Augmented Reality: A Guide to the Technologies, Applications, and Human Factors for AR and VR.* Addison-Wesley Professional, Sep 2016, ISBN 978-0134094236.

[9] Azad, P.; Gockel, T.; Dillmann, R.: *Computer Vision: Principles and Practice.* Elektor International Media, 2008, ISBN 9780905705712.

[10] Bradski, G.; Kaehler, A.: *Learning OpenCV: Computer Vision in C++ with the OpenCV Library.* O'Reilly Media, Inc., druhé vydání, 2013, ISBN 1449314651, 9781449314651.

[11] Cawood, S.; Fiala, M.: *Augmented Reality: A Practical Guide.* Pragmatic Bookshelf, Jan 2008, ISBN 978-1934356036.

[12] Dubrofsky, E.: *Homography Estimation.* Diplomová práce, The University of British Columbia, 2009.
URL
https://www.cs.ubc.ca/grads/resources/thesis/May09/Dubrofsky_Elan.pdf

[13] Gregory, J.: *Game Engine Architecture, Second Edition*. Natick, MA, USA: A. K. Peters, Ltd., druhé vydání, 2014, ISBN 1466560010, 9781466560017.

[14] Hartley, R.; Zisserman, A.: *Multiple View Geometry in Computer Vision*. New York, NY, USA: Cambridge University Press, druhé vydání, 2003, ISBN 0521540518.

[15] Havelka, J.: *Geometrické transformace obrazu*. Bakalářská práce, FIT VUT v Brně, 2008.
URL https://core.ac.uk/download/pdf/44391887.pdf

[16] Kipper, G.; Rampolla, J.: *Augmented Reality: An Emerging Technologies Guide to AR*. Syngress, Dec 2012, ISBN 978-1597497336.

[17] Loukotová, K.: *Současná uživatelská rozhraní dialogových informačních systémů*. Diplomová práce, Univerzita Karlova v Praze, Filozofická fakulta, Ústav informačních studií a knihovnictví, 2006.
URL https://is.cuni.cz/webapps/zzp/download/120065017

[18] Ottinger, T.; Langr, J.: *FURPS+*. 2009.
URL http://agileinaflash.blogspot.cz/2009/04/furps.html

[19] Peringer, P.: *Modelování a simulace: studijní opora do předmětu Modelování a simulace*. FIT VUT v Brně, 2012.

[20] Schmalstieg, D.; Höllerer, T.: *Augmented Reality Principles and Practice*. Pearson Education, Inc., Jun 2005, ISBN 978-0-321-88357-5.

[21] Shaer, O.; Hornecker, E.: *Tangible User Interfaces: Past, Present, and Future Directions. Foundations and Trends in Human–Computer Interaction*, ročník 3, č. 1–2, 2009: str. 1–137.

Příloha A

Obsah přiloženého CD

- doc – Technická zpráva a její zdrojový kód
- poster – Plakát
- src – Zdrojové soubory aplikace
- video – Prezentační video